View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by SJSU ScholarWorks

San Jose State University
SJSU ScholarWorks

Master's Projects | Master's Theses and Graduate Research

Fall 12-2010

# View Component of Web-based IDE to develop Web Applications in CakePHP

Swathi Vegesna
*San Jose State University*

Follow this and additional works at: https://scholarworks.sjsu.edu/etd_projects

Part of the Graphics and Human Computer Interfaces Commons, and the Software Engineering Commons

Recommended Citation
Vegesna, Swathi, "View Component of Web-based IDE to develop Web Applications in CakePHP" (2010). *Master's Projects*. 5.
DOI: https://doi.org/10.31979/etd.zhmc-jcs9
https://scholarworks.sjsu.edu/etd_projects/5

This Master's Project is brought to you for free and open access by the Master's Theses and Graduate Research at SJSU ScholarWorks. It has been accepted for inclusion in Master's Projects by an authorized administrator of SJSU ScholarWorks. For more information, please contact scholarworks@sjsu.edu.

# View Component of Web-based IDE to develop Web Applications in CakePHP

A Writing Project

Presented to

The Faculty of the Department of Computer Science

San José State University

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

By

Swathi Vegesna

December 2010

©2010

Swathi Vegesna

ALL RIGHTS RESERVED

SAN JOSÉ STATE UNIVERSITY

The Undersigned Writing Project Committee Approves the Writing Project Titled

View Component of Web-based IDE to develop Web Applications in CakePHP

By

Swathi Vegesna

APPROVED FOR THE DEPARTMENT OF COMPUTER SCIENCE

| | |
|---|---|
| Dr. Chris Pollett, Department of Computer Science | 12/13/2010 |
| Dr. Sami Khuri, Department of Computer Science | 12/13/2010 |
| Dr. T.Y. Lin, Department of Computer Science | 12/13/2010 |

# ACKNOWLEDGEMENTS

I am grateful to my project advisor, Dr. Chris Pollett for his great guidance and suggestions throughout the year. I would like to specially thank Dr. Sami Khuri and Dr. T.Y. Lin for being my committee members and for all their support and time. Mr. Sugi Widjaja, the developer of Model component of the IDE deserves special thanks for all his support and effort in developing the project. Also I am thankful for Ms. Tejasvi Palvai for encouraging me throughout the project and giving proper feedback. I extend my thanks to all my family and friends for being supportive in making this project successful.

# ABSTRACT

View Component of Web-based IDE to develop Web Applications in CakePHP

By

Swathi Vegesna

The aim of the project is to build a Web-based IDE (Integrated Development Environment) that enables users to create a Web application in PHP on the CakePHP framework. The view component of the IDE allows the users to build the CakePHP view templates, which include the HTML tags which can be properly linked with the controller functions. Users will be able to create the HTML form elements only by performing certain drags, drops and clicks. This project will create dynamic view elements, which is achieved by connecting them to the Controllers components. This IDE helps the users to create dynamic web pages even without good knowledge of HTML and server side scripting languages.

## TABLE OF CONTENTS

## List of Figures

# 1. Introduction

An Integrated Development Environment (IDE) is a software application that provides comprehensive facilities to computer programmers for software development. Some of the features of an IDE include; a source code editor, a compiler and/or an interpreter, a debugger and build automation tools. The current available popular IDE's available for PHP programmers are all Desktop-based and do not provide any frame work capabilities. The drawbacks of the Desktop-based IDE's are that the software should be installed and users cannot have remote access to their code. The current trend of web development is to follow a framework in developing rapid web applications so that the developer's life can be made easy in building consistent and more organized applications. The aim of this project is to build a Web-based IDE for PHP which enables developers to build web applications based on the CakePHP framework.

Desktop-based IDEs are traditional and the newly emerging are the Web-based IDEs. A Web-based IDE is a browser-based code development environment which can be accessed with a connection to the server and the available browser. This does not need any installation on the local developer workstation which in terms reduces hardware costs and management overhead can ensure everyone has the latest versions and patches. Consequently, developers can make code changes from anywhere, using any machine which has an internet connection and a browser. The IDE that is discussed in this report provides users with a text editor and various tools to ease the developer's work in building the complete web application in PHP.

The proposed project is to build an IDE that can enable developers to build web applications on CakePHP. CakePHP [1] is an open source web application framework which is ranked three [2] among the frameworks available in PHP, which follows Model-View-Controller

(MVC) architecture. The IDE developed provides the capability of building a project with all the necessary files and components to call it a complete web application in CakePHP. The framework provides an extensible architecture for developing with strict naming conventions and file structure. The projects that are built in this IDE are consistent and are very well organized which makes the development easier.

As CakePHP follows the Model-View-Controller (MVC) [3] architecture, the complete functionality of the IDE is categorized based on MVC architecture, an architectural pattern that isolates the domain logic from the user interface permitting independent development, testing and maintenance. The model component is the one that manages the data and the logic that manipulates it and responds to instructions from controllers. The controller component is the one that carries the complete business logic which receives the input and responses by instantiating the model objects. Finally the view component renders the models into suitable form for interaction. Implementing this design pattern eases development, deployment and maintenance.

The task of building this IDE is split based on the MVC architecture into two sub projects, Model component and View component. The Model component of the IDE is the one that enables users to create the database tables automatically and design complex queries to insert and retrieve data. The View component of the IDE is the one that provides the tools and functionality in building the views and linking the view elements with the controller functions. Users can create and develop the views or the CakePHP templates by dragging and dropping the elements and can also edit the code of the pages. The Model component was developed by Mr.

Sugi Widjaja, a Masters graduate student in Computer Science at San Jose State University and my part is to develop the View component of the Web-based IDE.

Being a Web-based IDE the complete project code resides on the server and the browser is the one that accesses the code and presents it to the developers. This IDE is designed for multiple users, a user can register and then login to his account to build and save his projects. All the user created projects are saved on the server and are accessible through any machine, when he logs into his account. All the Web application projects user creates can be previewed in the same browser on a specified URL (Uniform Resource Locator) to perform the relevant tests. Once a user creates a project he can always preview the application on a specified URL even without logging in unless, he needs to edit the application.

The proposed IDE is developed for the intermediate and advanced developers in building web applications based on CakePHP framework. The intermediate developer can use the design mode, where he can use the different tools and options to build the models and views. Edit mode was developed for an advanced developer to write in the code and save the content of the file. Specifically, for the building the views of the application, developers can use the design mode to insert, edit and position the HTML elements listed in the toolbar. In the edit mode, developers can edit the code and find the reflection in the design mode.

This report mainly concentrates on the complete feature set and implementation of the proposed IDE. Firstly, various softwares and tools used in building this IDE are discussed. Then, the report briefs about the preliminary work done as a part of initial research for the project during CS297, Spring 2010. The later section describes the architecture, folder structure and

features of the IDE and their implementation. This is followed by challenges faced and comparison with Phpanywhere, a web-based PHP IDE. The later sections describe the usability testing and the performance testing conducted on the IDE developed. The report ends with a conclusion and references.

## 2. Softwares and Tools

### 2.1 Apache 2.2.14

The main purpose of using Apache server is that IDE is web-based, which needs a server to host the application. All the requests from the browser are served by the Apache HTTP server commonly called as Apache server. A Web server [4] is a computer application that helps to deliver content that can be accessed through the internet. Apache is an open source web server primarily developed to serve static content and dynamic web pages on the World Wide Web (WWW). The Apache HTTP [5] Server Project is an effort to develop and maintain an open-source HTTP server for modern operating systems including UNIX and Windows NT. The goal of this project is to provide a secure, efficient and extensible server that provides HTTP services in sync with the current HTTP standards. When a user requests a web page on a web browser, a request is made to the server and it sends back the response to the browser.

### 2.2 MySQL 5.1.36

The Model component of the IDE, in simple words, is a mix of database and the business logic behind the data manipulation. MySQL is used as the back-end database for the proposed IDE. MySQL [6] is an open source relational database management system (RDBMS) that runs as a server providing multi-user access to a number of databases.

### 2.3 PHP 5.2.11

PHP [7] is a widely-used general-purpose scripting language that is especially suited for web development and can be embedded into HTML. It serves the purpose of developing dynamic web pages. This language consists of native API's to the Apache Server, so when the server gets a request for a web page with PHP, it calls the PHP interpreter to generate the

corresponding HTML and then this particular HTML page is returned to the client. The current project PHP 5.2.11 is used as it is compatible with the Apache 5.2.14.

### 2.4 phpMyAdmin

PhpMyAdmin [8] is an open source tool written in PHP intended to handle the administration of MySQL over the World Wide Web. It can perform various tasks such as creating, modifying or deleting databases, tables, fields or rows; executing SQL statements; or managing users and permissions.

### 2.5 CakePHP 1.3

The complete IDE developed is written in PHP, so we have chosen CakePHP as its framework. CakePHP [1] is a rapid development framework for PHP that provides an extensible architecture for developing, maintaining, and deploying applications. The framework uses the most common and famous design patterns like Model-View-Controller (MVC), Object-Relational Mapping (ORM), Active Record pattern and Front Controller pattern within the convention over the configuration paradigm. The framework is known for its capability in reducing the development costs and easing developers in writing less code. Also it turns the web application into a maintainable, modular and rapidly developed package. The framework also turns the web application more consistent and logical.

### 2.6 jQuery

The complete IDE is run on the browser, so all the tools and features that are developed are in JavaScript. jQuery [9] is a fast and concise JavaScript Library that simplifies HTML document traversing, event handling, animating, and Ajax interactions for rapid web development.

### 2.7 FireBug

FireBug is the most popular powerful browser plug-in for Morzilla FireFox. This tool allows us to inspect HTML, modify the style and layout in real-time, and provides the most advanced JavaScript debugger available. FireBug eases a developer's life of coding in JavaScript by providing perfect error messages and is the best debugging tool.

### 2.8 CKEditor

CKEditor [9] is an open source WYSIWYG text editor from CKSource that can be used in the web pages. The main aim of using this editor is to have a lightweight editor that requires no client-side installation. Also it is compatible with all the modern browsers as its code is written in JavaScript.

## 3. Preliminary work

Most of the initial research about the IDE was done during my CS297, Preparation for Writing Project in Spring 2010. During this research, I built the foundation required for developing the complete IDE like installing the softwares needed and setting up the server. I also performed some experiments to come up with the frameworks to be used. The complete work is categorized into four deliverables: first, Book Collection Website; second, Performance Tests on JavaScript Libraries; third, Drags and Drops; fourth, Layout of the IDE.

### 3.1 Book Collection Website

The main objective of this deliverable was to experiment with CakePHP and come up with a web application which uses the basic functionality provided by the framework. The goal of developing the Book Collection Website was to get hands on the MVC architecture and install all the required softwares needed to eventually develop the Web-based IDE. In this web application, registered users can login to their accounts to view the books available and add books to their cart for reservation.

Figure 1 shows the Book Collection page where users can view the entire collection of books and add books to the cart. The development of this web application gave me an oppurtunity to go through all the features provided by CakePHP and also install all the required softwares to host the Web-based IDE.

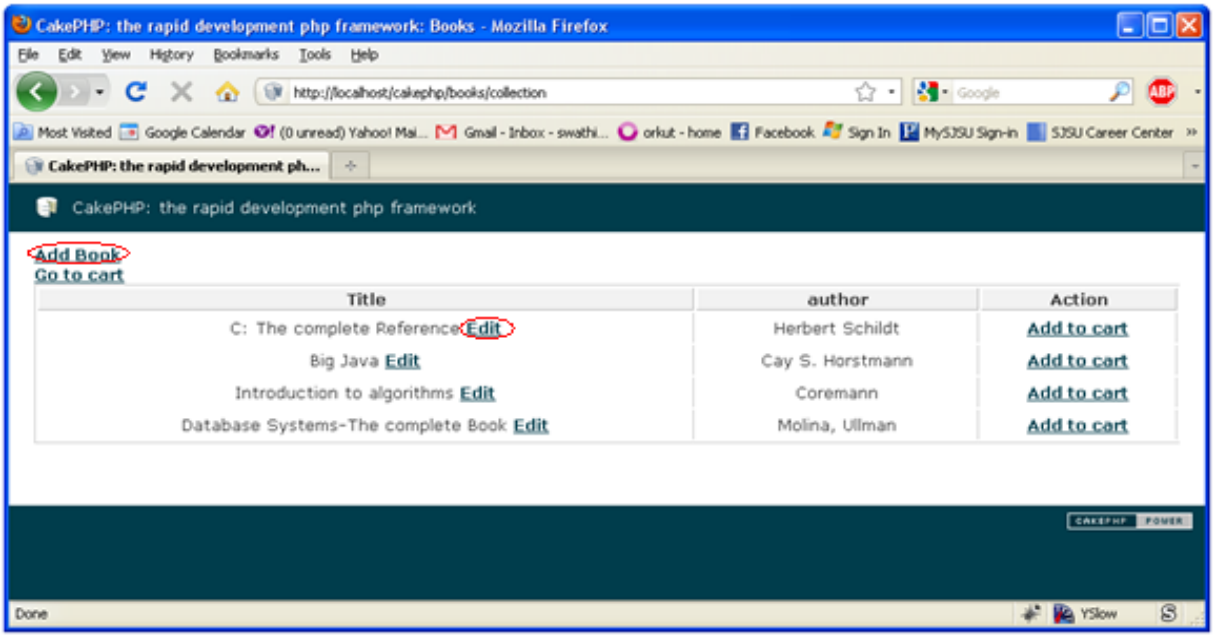



Figure 1. Book Collection page

### 3.2 Performance test on JavaScript frameworks

The main objective of this deliverable was to conduct different performance tests on the various JavaScript frameworks to decide on the framework to be used to build the GUI of the Web-based IDE. The selected four frameworks are namely, jQuery, YUI, DOJO and Prototype which were tested for their speed, weight and memory usage. The tools used for these tests are YSlow and Windows Task Manager.

| Frameworks | Grade | Performance score | HTTP requests | Weight |
|---|---|---|---|---|
| **jQuery** | A | 84 | 10 | 10.2K |
| **YUI** | B | 76 | 14 | 97.2 K |
| **DOJO** | A | 80 | 11 | 76.4 K |
| **Prototype** | B | 78 | 12 | 40.5 K |

Figure 2. Speed and Weight Test results

Figure 2 shows the results of the speed and weight test. Memory test was performed with the help of the Windows Task Manager and all the results proved that jQuery will be the most efficient JavaScript framework for the IDE.

### 3.3 Drag and Drops in jQuery

The main objective of this deliverable was to experiment with Drag and Drop functionalities in jQuery. The tools in the IDE should have these functionalities so that the view elements can be created only using some clicks, drags and drops. This experience in the drag and drops gave me the complete knowledge about the Draggable and Droppable classes in jQuery and the excellent features they offer which helped me in building the HTML elements in the toolbar.

### 3.4 Basic Layout of the IDE

The main objective of this deliverable was to build the basic GUI for the proposed Web-based IDE with a proper layout, all the navigation bars and folder structures. The complete layout of the IDE that is built is mainly categorized as Horizontal Navigation Bar, File system, View (Design and Edit modes) and HTML tool bar.

The basic layout is shown in the Figure 3, the top bar is the Horizontal Navigation bar, the left div is to list the different projects created by the user, the central view is the one where the text editor is placed and the right div is to show the tool bar for the View component and the suggestion bar for the Model component.

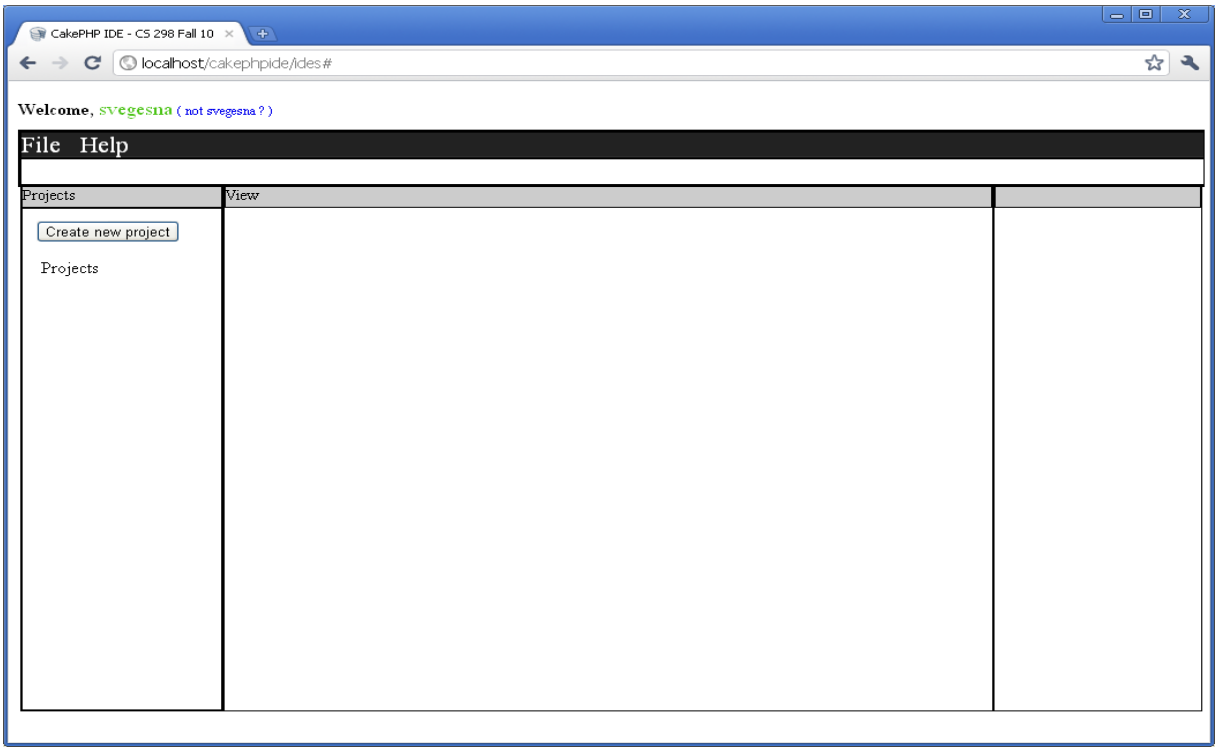



Figure 3. Basic layout of IDE

Overall, I learnt a lot from the preliminary work done in CS297 about CakePHP and jQuery libraries. All the effort helped in building the IDE successfully.

# 4. Web-based IDE

## 4.1 Architecture

The IDE was built to develop web applications on the CakePHP framework, which itself was built on the same CakePHP framework. The architecture of the project is the same as that of CakePHP shown in Figure 4 with the MVC pattern. Once the browser sends a request, the dispatcher routes it to the proper controller. The controller which has the business logic fetches the data from the model and sends the data back to the views which is shown by the browser.

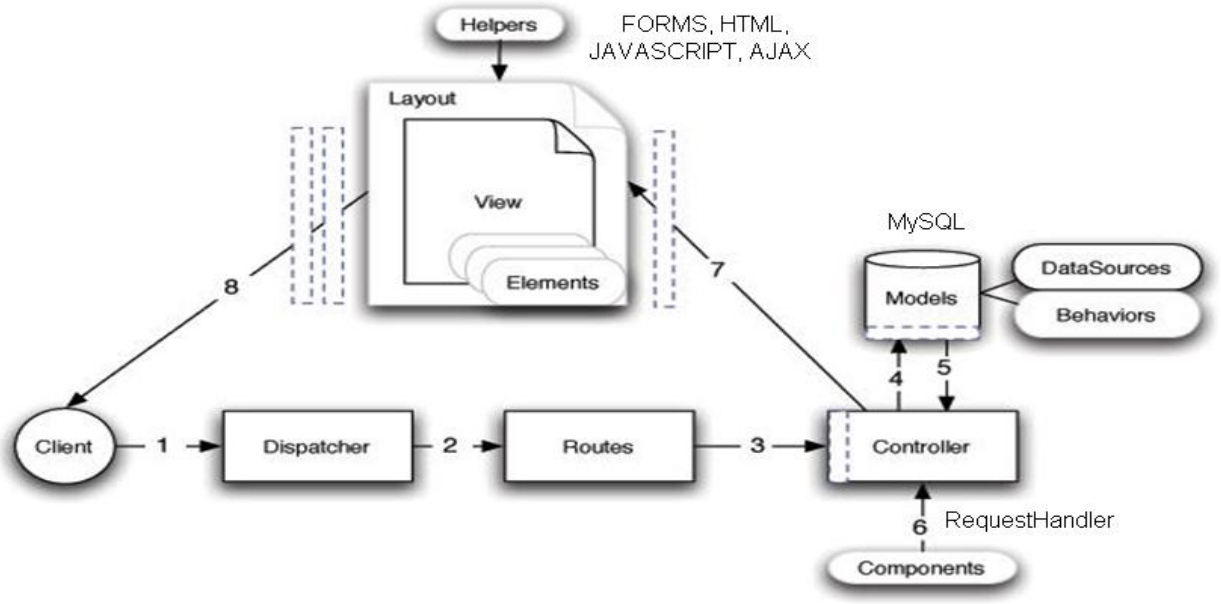



Figure 4. CakePHP Architecture

This architecture allows us to isolates the business logic with the presentation logic. Therefore, we can modify the look of our web page without changing the underlying business logic. As this pattern is implemented in our IDE, we can change the presentation of the web page without modifying any code used to deal with the database connection or the data manipulation or vice versa.

### 4.2 Folder Structure

CakePHP is a known for its folder structure and strict naming conventions, which keeps the web application consistent and well organized. Since the proposed IDE is developed on the same framework, the IDE gets the credit of being consistent. The main webroot folder of the server consists of 'cake' and 'cakephpide' folders, where 'cake' is the core CakePHP folder with all the functionalities developed by the CakePHP team and the later folder is where the Web-based IDE is built.

The folder structure of 'cakephpide' mostly interests the developers. One basic requirement for turning the 'cakephpide' into the web application is to set the constant variable 'CAKE_CORE_INCLUDE_PATH' to the path of 'cake' and 'ROOT' to the 'www' folder in WAMP (Windows Apache MySQL PHP) server or ' hotdocs' folder in Apache Server found in 'index.php' in 'webroot' folder of the application folder i.e. 'cakephpide' folder.

The main sub folders of the 'cakephpide' are namely

- Models - Contains the IDE's models.
- Controllers - Contains the IDE's controllers.
- Views - Contains all the presentational files of IDE (layouts, and view files).
- Webroot - Contains the folders for CSS style sheets, images, and JavaScript files.
- Config - Contains the configuration files (like database Config files).

#### 4.2.1 Naming Conventions

The naming conventions to be followed in CakePHP are

- Model class names should be singular and CamelCased.

  Ide → ide.php

- Controller class names should be plural, CamelCased, and end in Controller.

  IdesController → ides_controller.php

- Views are named after the controller functions they display (using underscores).

  getFile( ) →/cakephpide/views/ides/get_file.ctp

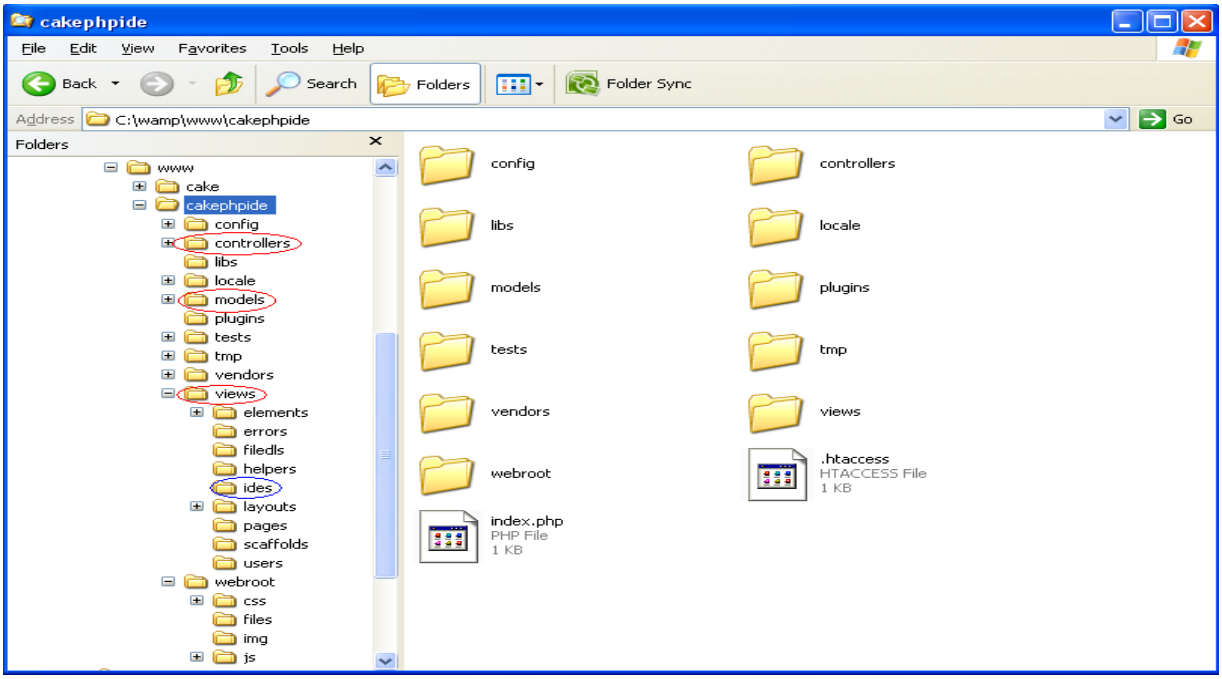



Figure 5. IDE's Folder Structure

Figure 5 shows the folder structure of the IDE, where all the model files placed in /cakephpide/models folder, controller files are placed in /cakephpide/controllers folder and view templates are placed in the /cakephpide/views/ides folder.

### 4.3 Features

This section describes the various functionalities and their implementation the IDE provides. The developed View Component of the Web-based IDE provides the following features: Multiple User with Multiple Projects capability, CakePHP based Web application creation, Tree view of the user's projects, Create and Edit views, Design and Edit modes of views, Form build Toolbar, Preview of Web application, and the Navigation bar.

#### 4.3.1 Multiple User with Multiple Projects capability

**Goal:** A Web-based IDE needs to have the capability of supporting multiple users. In order to build the capability, a login and registration should be built along with the IDE. Also, each user should be able to create as many numbers of projects desired, which is the multiple project creation capability.

**Implementation:** The implementation of this feature includes the following steps.

- Created a table named 'users' with the fields 'id,' 'username,' and 'password'
- /cakephpide/models/user.php → created a model class 'User,' and implemented validation rules for both empty fields and invalid username. The validation rule listed below is used for validating the login credentials. CakePHP provides excellent data validation rules, which makes a developer's life easy in validating the data.

```
var $loginvalidate= array(
        'username' => array('ifblank'=> array('rule'=> 'notEmpty', 'message' => 'Please enter your username'),
  'password' => array('ifblank'=>array('rule'=>'checkpassword', 'message'=> 'Please enter your password')),
);
```

- /cakephpide/controllers/users_controller.php → created a controller class UsersController, and implemented the enter( ) and register( ) methods
- /cakephpide/views/users/enter.ctp and register.ctp → the corresponding view templates have been created.

Once the user requests for URL for the IDE 'http://localhost/cakephpide/ides', the code is written in such a way that the page redirects to the Login page and if he is a new user he is taken to the Registration page. Once the user is done registering he needs to login again.

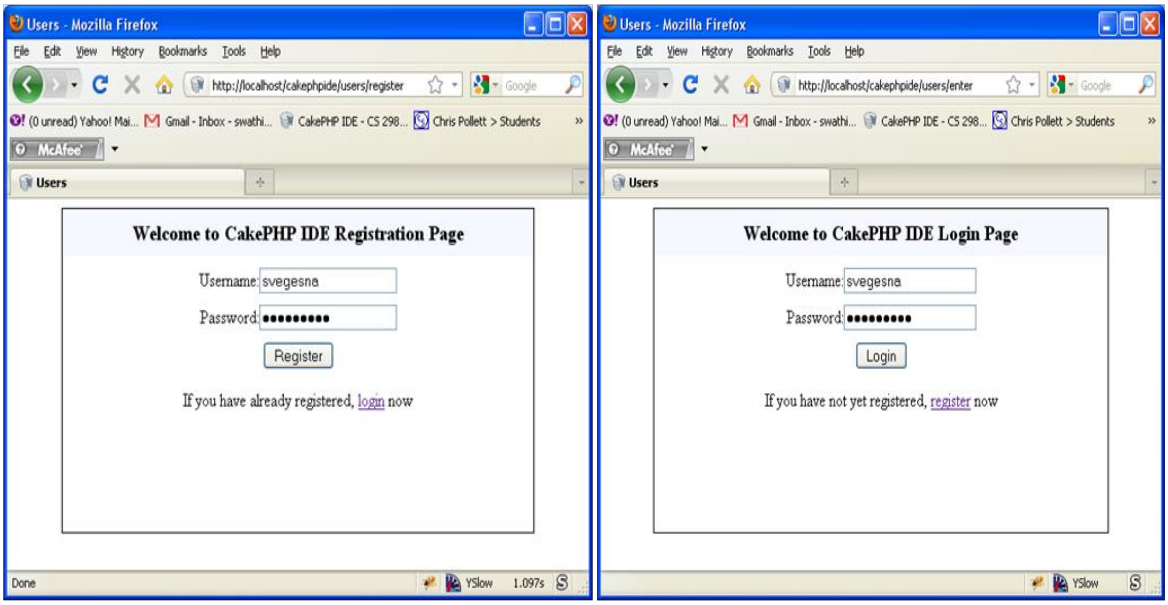



Figure 6. Registration and Login pages

Figure 6. shows the Registration and Login pages of the IDE developed, and Figure 7 shows the error messages thrown when there is data invalidation. Once a new user clicks 'Register' with proper details, a test for unique username is performed based on the results. The details are stored in the 'users' table and a new folder with folder name 'id' is created. All the projects the user creates are stored in this folder. The URL to preview his projects starts with 'http://localhost/userId/'.

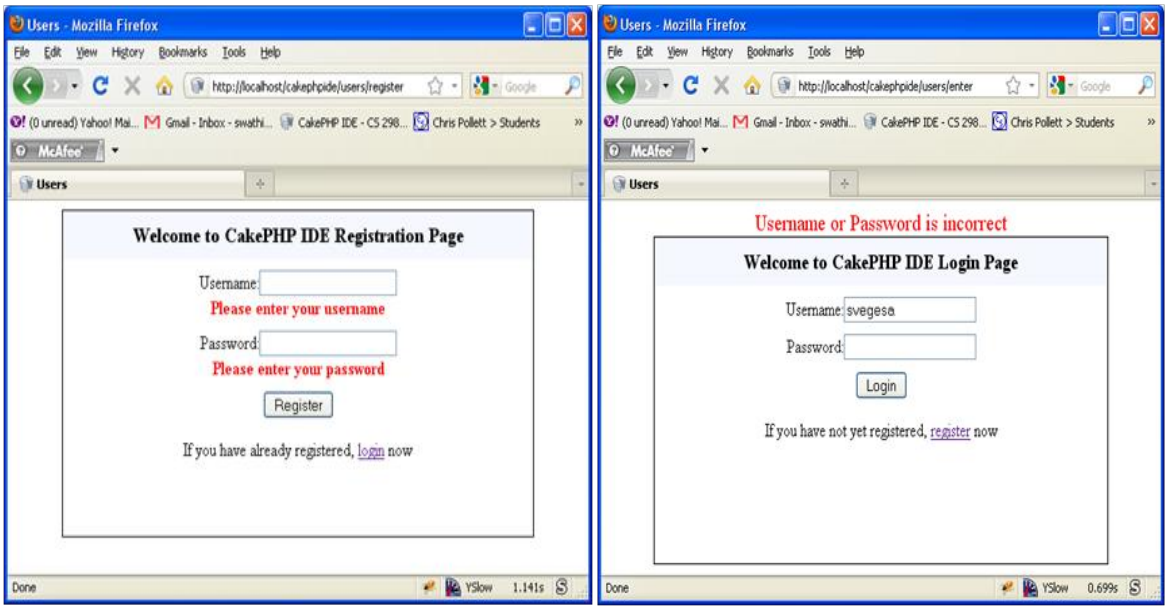



Figure 7. Registration and Login pages with error messages

This feature qualifies this IDE developed as a full blown web-based IDE, which can allow users to develop projects from anywhere or any machine.

### 4.3.2 CakePHP based Web application creation

**Goal:** This feature enables users to create projects based on the CakePHP framework, which makes this IDE outstanding as till-date there is no such Web-based IDE for PHP developed to build Web-applications on CakePHP.

**Implementation:** In order to create a Web application in CakePHP, the two main folders required are the 'cake' and the application folder named with the project name. As discussed in Section 4.2, the new application folder created should have proper path settings for both the constant variables 'CAKE_CORE_INCLUDE_PATH' and 'ROOT.' The creation of the new application folder is implemented in a method create( ) written in the controller class IdesController.

When the user clicks on the button 'create new project' a dialogue modal box appears asking for the project name, database, host name, username, and password for the database as shown in the

Figure 8. Once he enters all the required fields and clicks on the 'create' button an AJAX call is made to the 'create( ).' When the method executes, a new folder with the project name is created under the user's folder with all the file structure to be called as a web application on CakePHP. The following is the AJAX call using the jQuery's ajax( ) method.

```
$.ajax({'type' : 'POST',
        'url' : controller_url,
        data: 'projectname='+new_project_name +'&projectdb='+project_db +
              '&projecthost='+project_host+'&projectusername='+project_username+
              '&projectpassword='+project_password,
        success: function(msg) {
           $('#createprojectsuccessmsg').append('Successfully creating new project');
        }
        else  {
           $('#createprojectfailmsg').append('Failure in creating new project '+msg);
         }
});
```

Figure 8 shows the dialogue modal box that appears when the 'create new project' button is clicked. The 'create( )' method copies the basic application folder to the user's project location with the one all basic component files in each component. Also the associated database with proper credentials having one basic table is created. In each component a model, controller and view are also created.

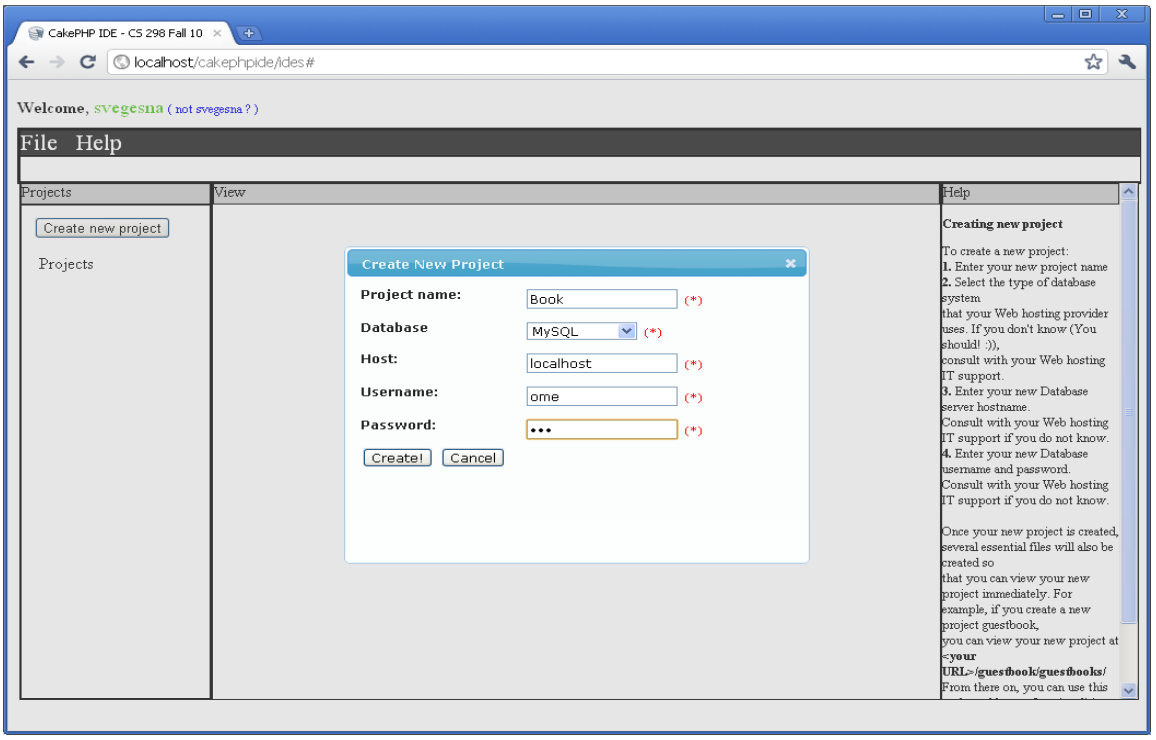



Figure 8. New project creation

All the basic requirements of a web application in CakePHP are set up for the user to edit the project. Similarly, a user can create any number of desired projects and start working on them.

### 4.3.3 Tree view of the user's projects

**Goal:** This feature is to show the key folders in the Web application in the Projects sections in a Tree view. The key folders are the Models, Controllers, and Views in the new project created by the user. The aim is to list all the files in each key folder and allowing users to edit them.

**Implementation:** Whenever a new project is created, the file-name and file-path details of the model, controller and the view files are stored in their respective database tables. In the left div of the IDE the Tree view of the folder structure is displayed. The following code in PHP will allow us to read the data from the data base tables and present it on the IDE.

```
foreach($project_data['views'] as $view_file){

        $this_view_id =  'view-'.$project_data['project_id'].'-'.$view_file['view_id'];

        printf("\$('div.view#%s').data('cfg', {'project_name'
:'%s','view_id':'%s','view_name'   : '%s' , 'view_filename':'%s' });",  $this_view_id,
$project_name, $view_file['view_id'], basename($view_file['view_filename'], '.ctp'),
addSlashes($view_file['view_filename']));

 }
```

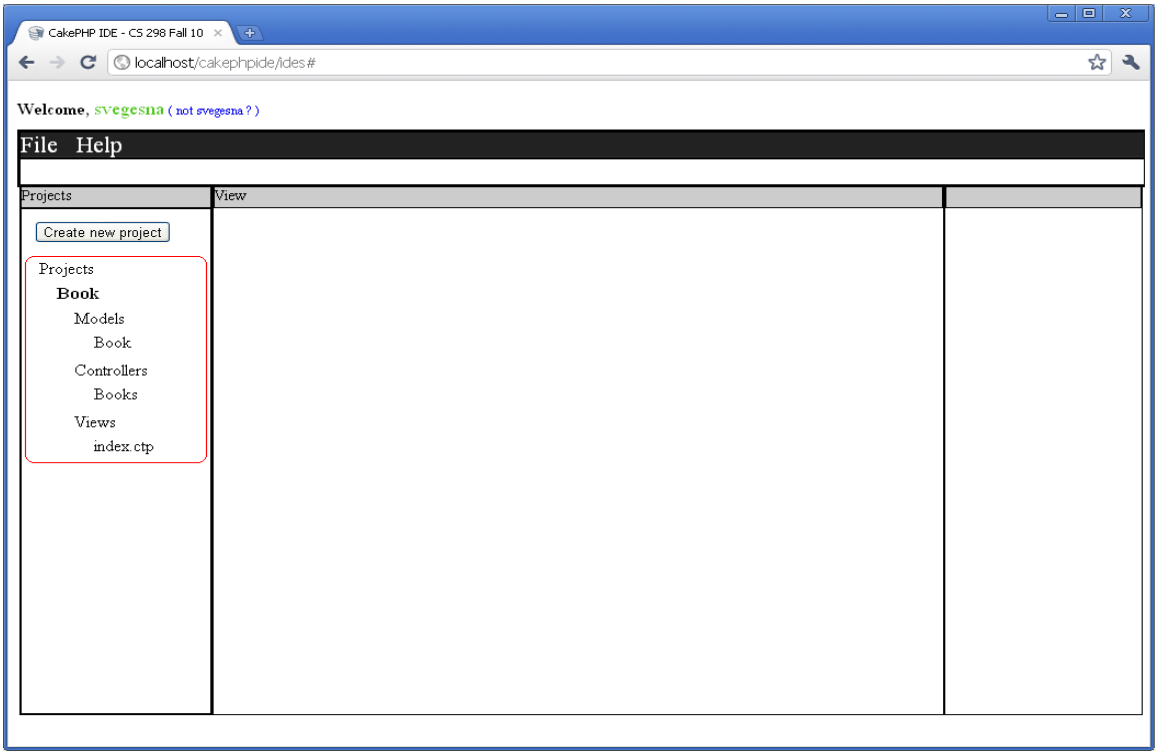



Figure 9. Tree view of the projects

Figure 9 shows the Tree view of the project created, with the model, controller, and view files.

### 4.3.4 Create views

**Goal:** This feature enables users to create additional views apart from the standard index.ctp when more controller functions are needed.

**Implementation:** The goal of this feature is to allow users to build more number of view components whenever they are desired. When the user right clicks on the 'Views' a context menu with an option of 'create new view appears'. When he clicks on the option a dialogue modal box appears asking for the name of the view as shown in the Figure 10. Once the user clicks on the create button, an empty view file with the given name is created in the views folder i.e, in "/userId/project/views/projects/new_view.ctp".

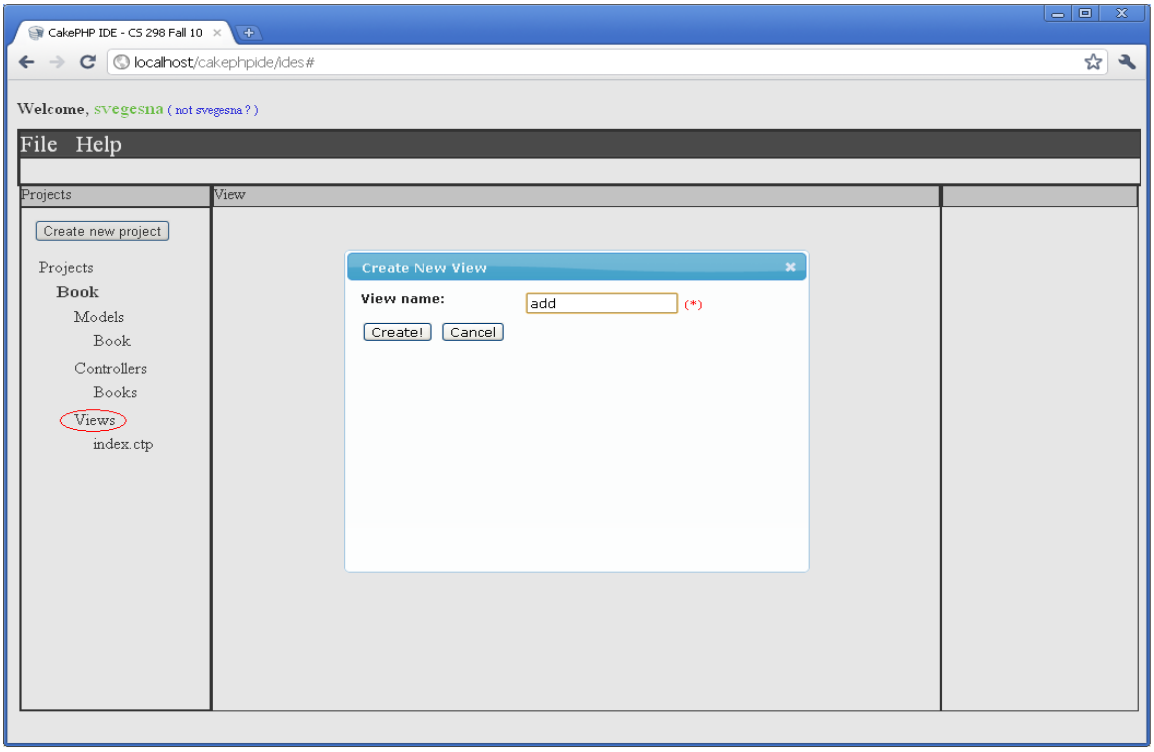



Figure 10. Creation of new view

The steps followed in creating the new a view are firstly, the database table holding all the view files is checked if the there are any name clashes. Secondly, if no matches are found a new view data is inserted into the table with the details like view name, project name and view path else it an error is shown that the view already exists. Finally a new file is created under the views folder.

### 4.3.5 Edit views

**Goal:** This is the best feature offered by the View component of the IDE, where users can edit the views using the Toolbar developed and also can view both their view templates in both design and edit modes.

**Implementation:** The view div in the center of the IDE pulls up the view template from the server, when the user clicks on the 'edit view' option in the context menu display on the right clicking the view files. This is achieved by a AJAX call through the controller method 'readviewfile( )'. The response of the AJAX call is the text form of complete code of the view file which is provided to the  in the design mode and CKEditor in the edit mode.

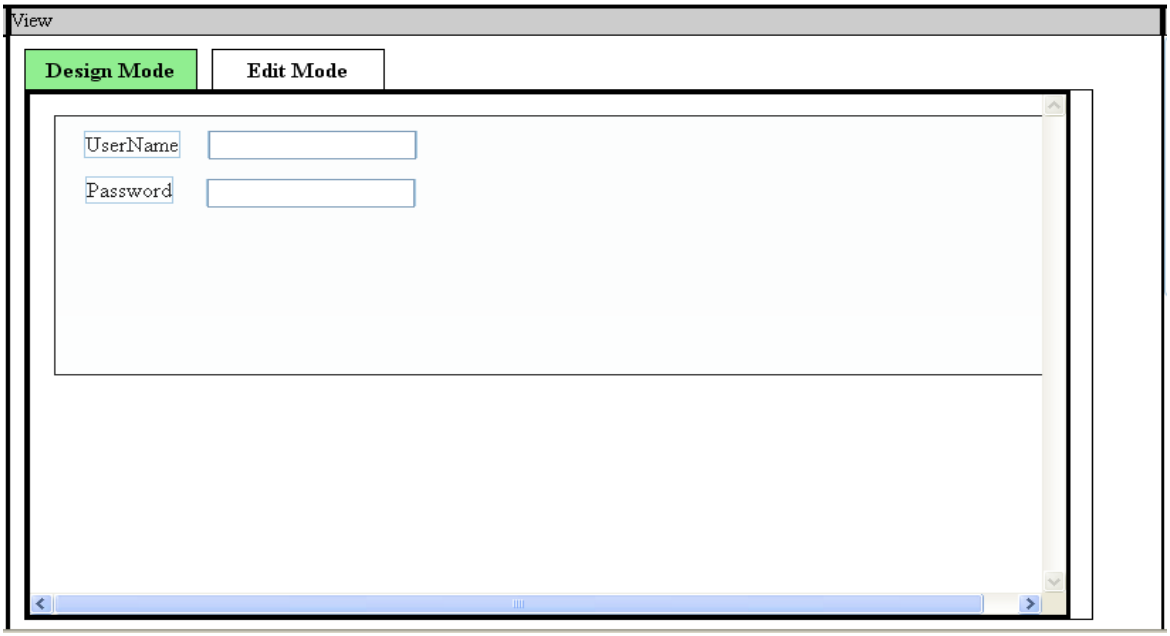



Figure 11. Design Mode

Figures 11 and 12 shows the view template in Design and Edit modes. CKEditor is embedded into the  of the Edit mode so that developers can write the code and save the changes made.

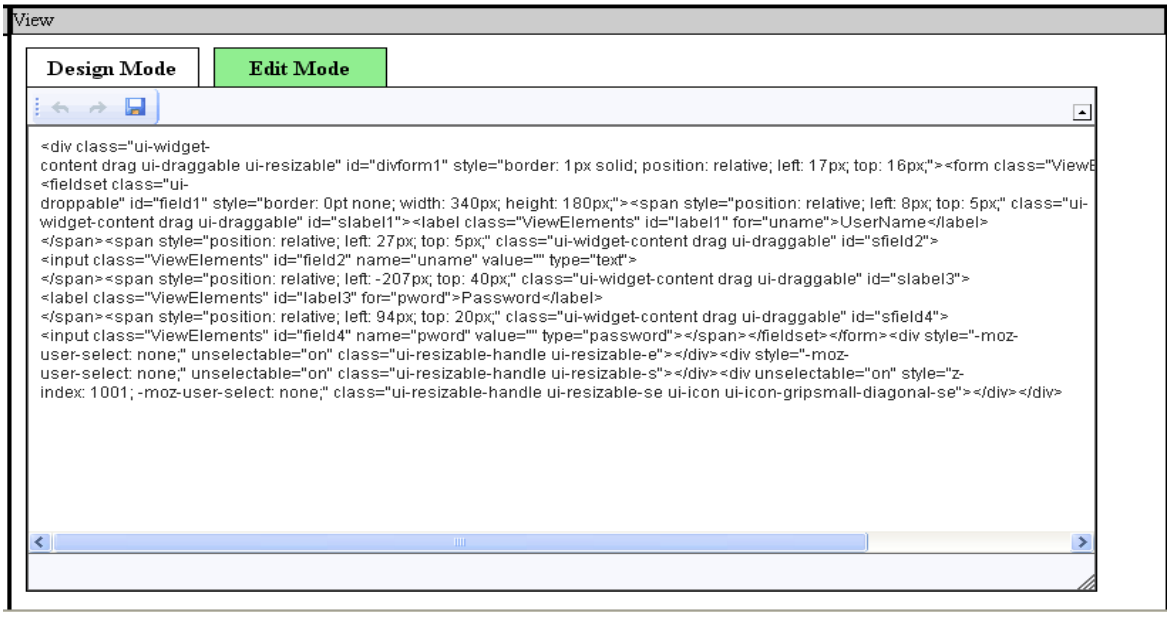



Figure 12. Edit Mode

### 4.3.6 Form Build Toolbar

**Goal:** The Form build Toolbar provides users with an easy way to build the Forms in the view component and link them to the controller functions.

**Implementation:** The Form build Toolbar appears on the right side of the view div where the whenever a view template is edited. Figure 13 shows the list of elements in the toolbar. In the 'Elements' tab all the form elements are listed and in the 'Attributes' tab all the various attributes for each element is listed for developers to beforehand what are the values needed to be changed on creating them. As this toolbar is specific to build a Form, the list of form element appears only when user needs to build a Form. The functionality of these elements is so implemented that, all these elements are draggable on to the Design mode to build the Form.

```
$(".formelements").draggable({
        helper:'clone',
        cursor: 'move'
});
```

In this report a use case of how to create a form and create a label are discussed. The user need to drag the 'View form' tag on to the Design mode on the View div. An empty form is created which can be draggable and resizable.

The following is the JavaScript code which does the job of creating the draggable and droppable elements of the respective Form elements and the view div.

```
$(".viewFile").droppable({
        accept: '.formelements',
        drop: function(event, ui) {
        $('.viewFile').append(form);
        $(".drag").draggable({
                cursor: 'move',
                containment: "parent"
        });
        append_rightClick();
        $("#divform"+formId).resizable();
        $("#field"+formId).droppable({
                accept: '.formElements',
                drop: function(event, ui){
                        addElement(d, fId);
                        append_rightClick();
                }
        });
        }
});
```

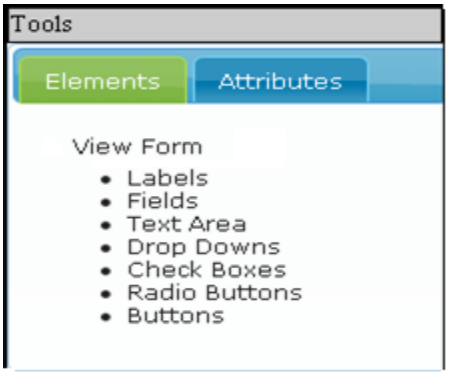



Figure 13. Form Build Toolbar

Once the Form is ready, user can start building the Form by populating it with different form elements listed in the toolbar. When the user can drags the 'Labels' tag and drops it on to the Form a default label along with a default text field get appended to the form, as shown in the Figure 14. In order to edit the default elements, user needs to right click on the element to be edited and click on the 'Edit' option in the context menu.

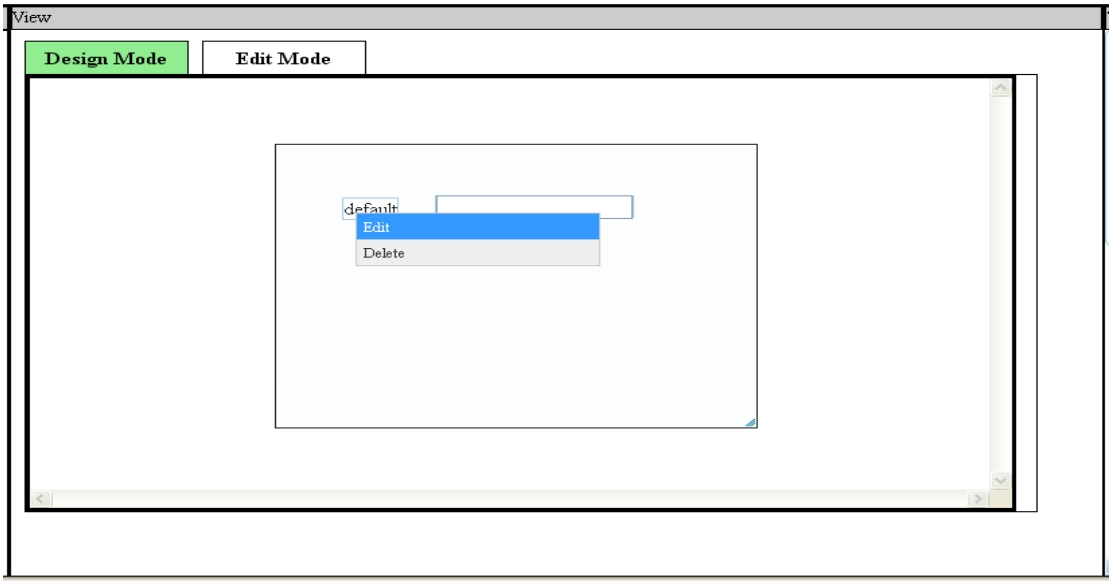



Figure 14. Right Click Implementation on View element

A dialogue modal box appears where user needs to enter the details as shown in the Figure 15. Once the user clicks on 'OK' button the requested change is made and the label gets updated as shown in the Figure 16.

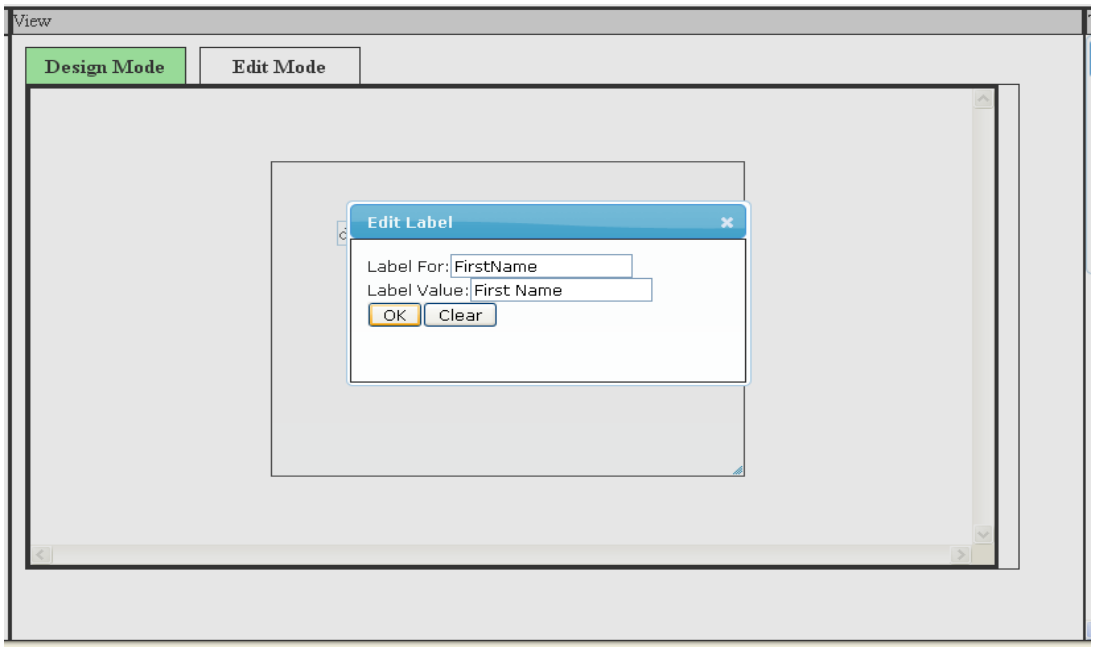



Figure 15. Editing a Label

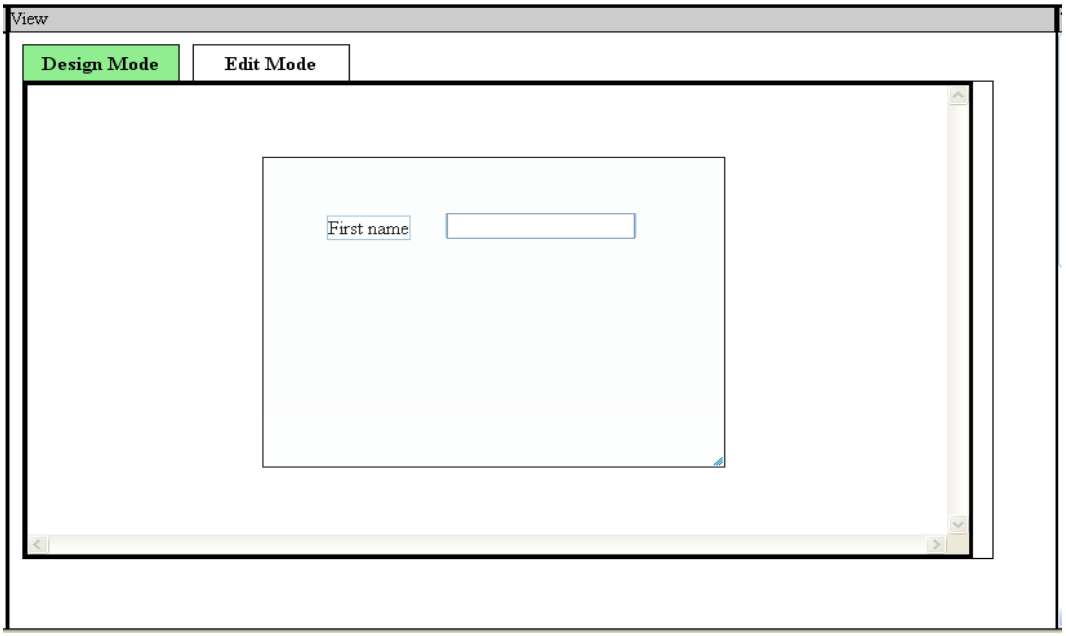



Figure 16. Updated Label

Similarly, user can create Textareas, Check buttons, Radio buttons, Text fields and Buttons on the Form. Once he is done adding all the form elements user can right click on the form and edit the form. A modal dialogue box shows up asking for the type of the 'method' either 'get' or

'post' and the option list of all the controller functions to which the Form data can be submitted. All the functions listed in the option list are obtained from the controller file and are populated when a view file is edited. The following is the code snippet used for fetching the functions from the controller file. PHP reflections are effectively used in order to get the controller methods from the associated controller files.

```
function getformfunctions(){
		$file_path = $this->params['form']['component_filename'];
		$project_name = $this->params['form']['project'];
		$component_type = $this->params['form']['component_type'];
		$component_name = $this->params['form']['project']."s";
		$cfile_path = sprintf("%s%s%s%s%s%scontrollers%s%s_controller.php",
CAKE_CORE_INCLUDE_PATH, DS, $uid,DS, $project_name, DS, DS,
Inflector::underscore($component_name));
		$all_contents = file($cfile_path);
		$obj = $this->instantiate_model($project_name, $model_name, $model_filename);
		if($obj){
			$results = get_class_methods(get_class($obj));
		}
		print(json_encode($results));
}
```

Once the user selects the proper options and clicks on 'OK' button, at the backend the form is linked up with the controller function. This is how a Form build Toolbar eases the creation of Forms in the view templates.

### 4.3.7 Preview of Web application

**Goal:** This feature allows developers to preview their web application developed on a different window. This helps in conducting various tests on the application they built.

**Implementation:** To be consistent in feature implementation, right clicking on the project name will display a content menu with options of Export and Preview. When the user selects 'Preview' a new window opens with the URL of web application developed. The following JavaScript code can perform the above job.

```
function popitup(url) {
   newwindow=window.open(url,'name','height=350,width=400');
   if (window.focus) {newwindow.focus()}
   return false;
 }
```

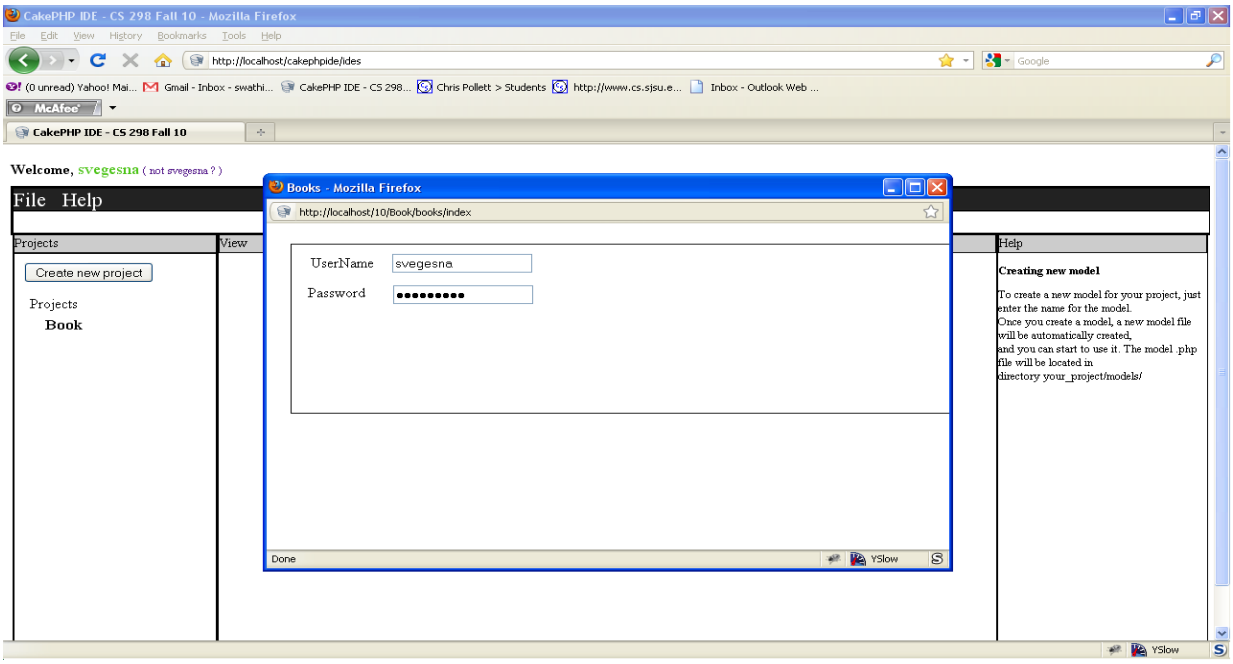



Figure 17. Preview of the application

Figure 12. shows the new window with the preview of the web application. This can be one feature developers really would appreciate as no other IDE provides this functionality.

### 4.3.8 Navigation bar

**Goal:** One feature every IDE should have is a navigation bar on the very top of the page, which can have functionalities of creating new projects and getting info about the IDE.

**Implementation:** In order to be simple the only two drop downs are implemented in the Navigation bar. One is the 'File' with a dropdown list containing 'New project' and 'Log out'. The functionality behind the 'New project' is same as the creating new project. The functionality of 'Log out' is to clear the session variable and logout of the IDE. If the user wants to access the IDE, he needs to login back.

The implementation of the above functionality of logout can be achieved by the following code in the controller file.

```
function logout()  {
   $this->Session->destroy();
   $this->redirect(array('controller' => 'users', 'action' => 'enter'));
 }
```

The 'Help' dropdown contains the 'About' link which opens a window giving info about the Web-based IDE. Figure shows the new window that opens when the 'About' link is clicked.

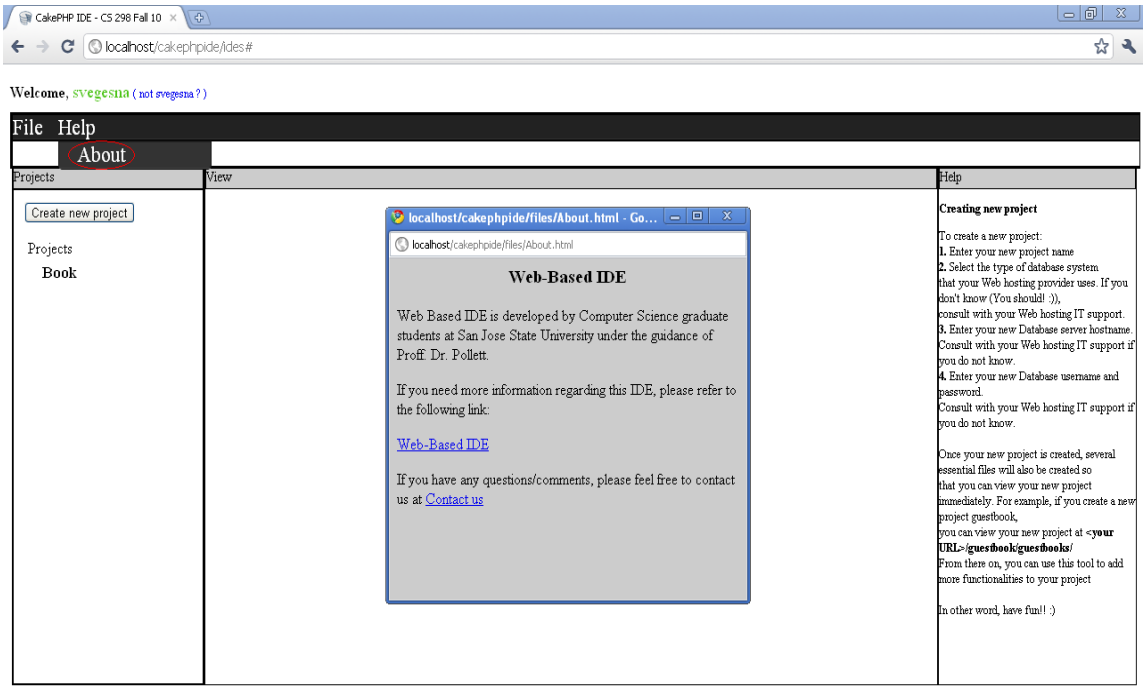



Figure 18. Help –About Window

## 5. Challenges faced

This section describes the various challenges and issues faced during the development of the Web-based IDE. The major challenges faced include View-Controller Interactivity, Version compatibility, Browser compatibility, and Operating compatibility issues.

### 5.1 View-Controller Interactivity

Implementing the View-controller interactivity is one of the biggest challenges faced in building the View component of the IDE. This feature makes the IDE outstanding as it allows to linkup the view Form element with the controller functions. User is given an option list of all the controller functions to which the Form data can be submitted. This is achieved by parsing the controller file to get all the functions names and according edit the Form action. The parsing was initially done using regular expressions which is quite tedious as most of the time it failed giving wrong results. This solved by using reflections in PHP and the other functionality provided by CakePHP.

### 5.2 Version compatibility Issues

Choosing the proper versions of the softwares is one of the biggest challenges faced. Not all the Apache HTTP server versions are compatible with all the available versions of PHP. With the release of the latest versions every six months, the foundation of softwares built during CS297 did not help much and so all the foundation had to be rebuilt. The latest version of Apache is not compatible with the one in PHP. This issue was solved by a trial and error and testing for the version of PHP, which is compatible with the latest version of Apache server. As the latest version of CakePHP was released the complete code base had to be upgraded so that it is compatible with version 1.3.

### 5.3 Browser compatibility Issues

The most common issue every Web application faces are the browser compatibility issues. The initial plan of using HTML5 has been dropped because not all the browsers are supporting 100% of it and have been avoiding usage of the cutting edge features of the language. The layout of the IDE is so built to avoid the different screen sizes and get the same feel of the IDE in all the modern browsers, such as Morzilla Firefox, Google chrome, Safari, and Internet Explorer 8. Also the CSS compatibility is taken into consideration, so the CSS features Internet Explorer does not support are avoided.

### 5.4 Operating System compatibility Issues

The Model component of the IDE is developed in MAC operating system, whereas the View component is developed in the Windows environment. Integrating the code and coming up with the perfectly working IDE was the biggest challenge faced. Many changes in the code were done, some of which included adding additional functions to support the file path in both Windows and Mac and changing the file structure to have a common platform. After solving all these issues, the current codebase is compatible with both Mac and Windows operating systems.

## 6. Comparison with phpanywhere

Phpanywhere [12] is a Web-based free IDE or application that gives developers all the code editing capabilities they need to develop PHP applications online. The following is the feature comparison of the proposed IDE and phpanywhere.

| Features | Proposed IDE | Phpanywhere |
|---|---|---|
| MVC pattern | Implemented | N/A |
| Framework | CakePHP | N/A |
| Tools to build Views | Available | N/A |
| Preview | Available | N/A |
| Tabs | N/A | Available |
| Code Indentation | N/A | Available |
| Syntax highlighting | N/A | Available |

Phpanywhere is a Web-based IDE very much similar to the Desktop-based Eclipse IDE. When it comes to the performance both the IDEs are equally faster as they retrieve the files from the server without much of a wait. The main advantage of the proposed IDE over Phpanywhere is, the framework used and the development of web applications in CakePHP. The main drawbacks are lack of syntax highlighting and code indentation which are much related to the editor component of the IDE. But the Tools provided by the IDE and the functionality they offer, mostly prevents the use of the text editor. The Toolbar provided makes the IDE developed outstanding in comparison with Phpanywhere.

## 7. Usability Testing

Usability testing is one such testing used to evaluate the product developed by testing it on users. The developed IDE is tested for its usability or ease of use. The following is the feedback given by three CS major graduates from San Jose State University on the View component developed.

**Madhuri Gollu:**

The user interface and the layout of the IDE is very user friendly and had some similarities with the Desktop-based Eclipse IDE. This similarity helps in getting acquainted to the IDE much faster. The IDE's performance is found to be very fast as there is no sluggish behavior in the cursor movement. The IDE is found to be very useful in creating view templates, the suggestion I give is to change the display of the form elements and make them visible only when a form is dragged.

**Sowmya Sampath:**

The toolbar for the View component can be called as the best feature provided by the IDE, as it makes the creation of a form very easy. The drag and drop mechanism for building the view elements makes this IDE outstanding from other IDEs. The right click implementation is found very consistent which makes it very user friendly. I would like to suggest having a help menu in the toolbar to give the user directions of how he can create and edit the form .Overall this IDE eases development phase in building the web applications in CakePHP and can be considered as the basic foundation of an IDE to build a very complicated IDE.

**Priya Gangaraju:**

The most challenging and best part about the IDE is the View-Controller interactivity; this is not found in any of the existing IDEs. Giving an option of linking the form elements with the

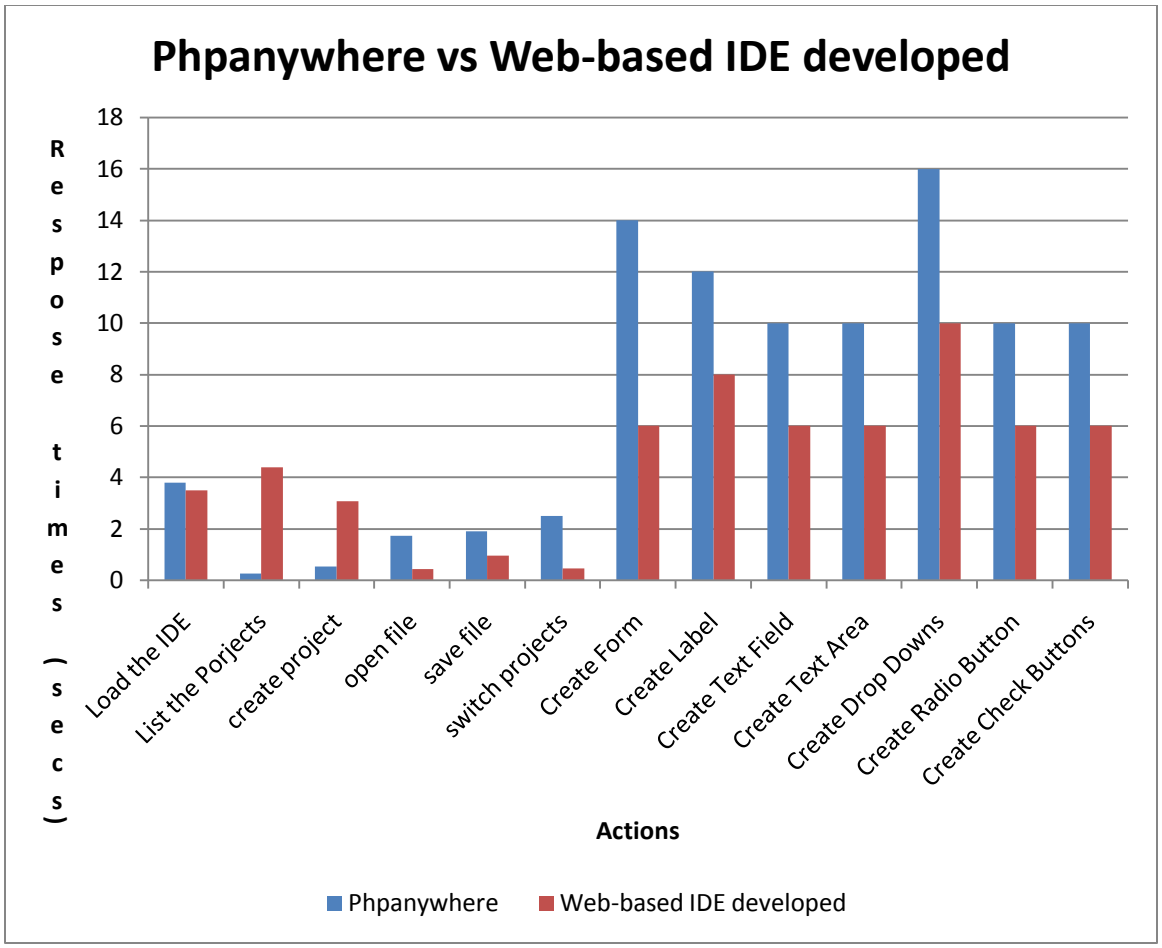
controller function must be a real challenge. Also the Preview feature provided makes it real useful for testing the developed applications. I would like have a basic CSS to be provided by the IDE to make the basic template created to be well presented. Overall the IDE built is very useful.

**Response to above Feedback:**

- The Toolbar is changed in a way to display the form elements only when a form is dragged on to the view div.
- A help menu is included in the bottom of the Toolbar.
- A generic CSS file is included to improve the presentation of the basic view templates.

## 8. Performance testing

The performance testing is used to evaluate the speed or effectiveness of the application developed. This is conducted by measuring the response times of each functionality developed.




The above chart shows the response times of different actions performed on both Phpanywhere and the Web-based IDE. In more than 80% of the jobs, the Web-based IDE proved to be the fastest IDE. From these observations we can conclude that with the implementation of various features like View-controller interactivity and Form build Toolbar, the IDE developed is much efficient in building the Web applications.

## 9. Conclusion

The Web-based IDE developed will allow users to build rapid wed applications on CakePHP framework. The various tools in the IDE simplify the development phase of the CakePHP applications. The View-Controller interactivity and Form build Toolbar are the key features which makes IDE very useful in building the CakePHP view templates. These features make it unique as there are only few Web-based IDEs for PHP like Phpanywhere and there are no available IDEs for building web applications on CakePHP. The performance testing performed on the IDE, gives a proper evidence for the effectiveness and efficiency of the IDE. Also from the feedback given by the CS graduates, it is evident that PHP developers will like to use the IDE in building web applications in CakePHP framework. Overall, the Web-based IDE developed gave me an opportunity to explore PHP and new development frameworks like CakePHP in PHP and jQuery in JavaScript. In the development process, I have gained an extensive knowledge of how to create web applications on the MVC web frameworks. This also helped me in getting hands on building rapid and interactive web applications.

## References


[1] CakePHP. Retrieved November 06, 2010, from http://cakephp.org/

[2] Top 10 Ranking PHP Frameworks. Retrieved November 14, 2010 from http://www.phpframeworks.com/top-10-php-frameworks/

[3] Model–View–Controller. Retrieved November 06, 2010 from http://en.wikipedia.org/wiki/Model%E2%80%93View%E2%80%93Controller

[4] Web Server. Retrieved November 06, 2010 from http://en.wikipedia.org/wiki/Web_server

[5] Apache HTTP server. Retrieved November 06, 2010 from http://httpd.apache.org/

[6] MySQL. Retrieved November 06, 2010 from http://en.wikipedia.org/wiki/MySQL

[7] PHP. Retrieved November 06, 2010 from http://www.php.net/

[8] PhpMyAdmin. Retrieved November 06, 2010 from http://www.phpmyadmin.net/home_page/index.php

[9] JQuery. Retrieved November 06, 2010 from http://jquery.com/

[10] FireBug. Retrieved November 06, 2010 from http://getfirebug.com/

[11] CKEditor. Retrieved November 06, 2010 from http://ckeditor.com/

[12] Phpanywhere. Retrieved November 20, 2010 from http://phpanywhere.net/overview